REMARKS

Prior art rejections

Claim 1 is an independent claim and has been rejected under 35 USC 102(e) as being anticipated by Edwards (5,901,315). Applicant has substantially amended claim 1, and submits that claim 1 as amended is patentable over Edwards, either alone or in view of Radigan. A number of limitations of claim 1 are in particular discussed individually. For each limitation, Applicant discusses why the limitation in question is not unpatentable over Edwards alone or in view of Radigan.

(1) High-level language of probe program different than that of base program

The claimed invention is limited to a base program and a source program. The probe program is generated from source code written in a high-level language. Importantly, "the high-level language of the source code from which the probe program is generated [is] different than a high-level language in which the base program is written." (See, e.g., para. [0025] of the patent application as filed, in which it is stated that the "HLL in which the base program 102 is written may be the same or different than the HLL in which the base program 104 is written.")

In Edwards, the target or base program is written in Java and also optionally in C/C++. That is, there is a

> "target" application that is written in Java but that optionally comprises one or more native method dynamic load libraries or "dll's." Typically, a dll is written in "C" or "C++" programming language, and thus it will be appreciated that the present invention provides a mechanism for simultaneously debugging Java and C/C++ code."
>
> . . . .
>
> One of the features of the present invention is the ability to debug native methods (C/C++ 16- or 32-bit code in a .dll) from the target Java application. Thus, according to the present invention, a target application may comprise both Java code and native method dll's.

(Col. 3, ll. 53-61; col. 4, ll. 39-43)

Likewise in Edwards, the probe program is written in Java, and also optionally in C/C++ where the target or base program is also written in C/C++.

> Thus, according to the present invention, a target application may comprise both Java code and native method dll's. To meet this goal, the probe 24 (of FIG. 2) preferably is a mixture of both C and Java code. The component that consists of the "C" code is the jademon process 34 of FIG. 3, and the component that consists of the Java code is the javaprobe process 36. As will be seen, the jademon process uses the system debug API . . . the control and manipulate the "C" or "C++" portions of the application under debug. Likewise the javaprobe process 36 uses the Java debug API . . . to control and manipulate the Java portions of the application under debug.

(Col. 4, ll. 41-51.)

Therefore, in Edwards, if the target program is written in Java, then the probe program is also written in Java, because the Java portion of the probe program is used to debug the Java portion of the target program. If the target program in Edwards is also written in C/C++, then the probe program is also written in C/C++, because the C/C++ of the probe program is used to debug the C/C++ portion of the target program. This is different than in the claimed invention, in which the high-level language (e.g., C/C++, Java, etc.) of the source code of the probe program is different than the high-level language in which the base or target program is written. For this reason alone, Edwards does not render claim 1 unpatentable.

Applicant has also reviewed Radigan and the other cited references in detail, and cannot locate any other reference as disclosing this aspect of claim 1. Therefore, the claimed invention is further not obvious over Edwards in view of Radigan or any other cited reference.

(2) Source code of probe program compiled by compiler specific to probe program

The claimed invention is further limited to "the source code from which the probe program is generated being compiled by a compiler specific to compiling probe programs like the probe program and not for compiling the base program." That is, as noted in the patent

application as filed, "the HLL compiler 112 is preferably specifically employed for compiling the probe program 102, and not necessarily for compiling the base program 104." (Para. [0025].) Likewise, the "HLL compiler 112 preferably [is] specifically intended for compiling probe program source code into probe programs," and this HLL compiler 112 "compiles the HLL probe program source code 114 into the executable probe program 102." (Id.)

By comparison, in Edwards, there is no compiler identified that compiles the high-level language of the source code of the probe program, let alone a compiler that is "specific to compiling probe programs like the probe program and not for compiling the base program." Applicant has further reviewed the other cited references, and cannot locate any compiler that is specifically for compiling probe programs and not for compiling the base or target program. That is, in the claimed invention, there is a special type of compiler that is just for compiling probe programs like the probe program recited in claim 1. Such a special type of compiler is not disclosed within the prior art, and for this reason as well claim 1 is patentable over Edwards alone or Edwards in view of any other cited reference.

*(3) Probe program independent of processor architecture "**due to . . .**"*

Claim 1 is further limited to "the probe program being independent of an architecture of the one or more processors *due to* the probe program being initially written in the high-level language *and* compiled by the compiler." (See para. [0027] of the patent application as filed.) That is, the probe program is independent of the processor architecture for two reasons: first, the probe program is initially written in a high-level language; second, this high-level language is compiled by the compiler that has been discussed above.

By comparison, in Edwards, the probe program is independent of the processor architecture solely because it is at least in part written in the high-level language Java, as noted by the Examiner on page 5 of the most recent office action. That is, the probe program is not independent of the processor architecture also because this high-level language of the probe

program is compiled by a compiler. No other cited reference discloses this aspect of the claimed reference either, especially in relation to the high-level language Java of the probe program. For this reason alone, too, claim 1 is patentable over Edwards alone or in combination with one or more other references.

(4) Probe program independent of machine code representation of base program

Claim 1 has been amended so that "the probe program [is] independent of a machine code representation of the base program," where "the machine code representation of the base program [is] tied to an instruction set of the one or more processors." (See para. [0027] of the patent application as filed.) By comparison, in Edwards, as has been discussed above, we have the following situations (see (1) above in particular). First, if the base program has a portion written in Java, then the probe program also has to have a portion written in Java, which means that there is no "machine code representation" of the base program, because Java is processor architecture independent, as noted by the Examiner on page 5 of the most recent office action.

Second, if the base program in Edwards has a portion that is initially written in C/C++, then the probe program also has to have a portion written in C/C++. Inasmuch as this means that the base program has a machine code representation tied to the instruction set of the processors, then it logically follows that the probe program also has a machine code representation that is tied to the instruction set of the processors. As such, the probe program is not independent of the machine code representation of the base program in Edwards.

No other cited reference discloses this aspect of the claimed reference either. For this reason alone as well, therefore, claim 1 is patentable over Edwards alone or in combination with one or more other references.

(5) Abstract syntax tree having particularly limited nodes

Claim 1 has also been amended to recite "an abstract syntax tree (AST) having a plurality of nodes." Importantly, "at least some of the nodes of the AST represent[] objects of the base program," while "other of the nodes of the AST represent[] objects of the probe program." As such, "a first address space of the objects of the base program and a second address space of the objects of the probe program are switched between by traversing the AST." Applicant notes that this added language in independent claim 1 originates from the originally presented language in independent claim 14.

Edwards in view of Radigan does not teach, disclose, or suggest such a particularly limited abstract syntax tree (AST). While Radigan does disclose an abstract syntax tree, as noted by the Examiner on page 11 of the most recent office action, it does not disclose an AST that has some nodes representing objects of the base program, and other nodes representing objects of the probe program. Rather, the compiler program 140 in Radigan "parses source program 130 to an annotated intermediate language program 150 . . . compris[ing] . . . a parse tree" (col. 5, ll. 40-48) which is a type of AST. In this manner, the AST in Radigan is used as is predictably encountered within the prior art: for representing a single computer program.

If an AST is extended in Edwards in view of Radigan to two computer programs – a base program and a target program as in the claimed invention – then the predictable and prior art usage of such an AST suggests that there would be two AST's: a first AST for the base program, and a second AST for the target program. By comparison, the claimed invention employs an AST other than the manner in which it is normally and predictably used within the prior art – by having one AST that has some nodes corresponding to the base program, and other nodes corresponding to the probe programs. In this way, an AST is novelly used in the invention to switch between the address space of the base program and the address space of probe program. By comparison, Edwards in view of Radigan does not teach, disclose, or suggest such a novel usage of an abstract syntax tree.

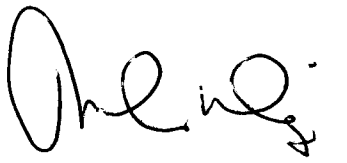
That is, there is no teaching, suggestion, or disclose within the prior art to (1) employ an abstract syntax tree encompassing two computer programs; and, (2) employ an abstract syntax tree encompassing two computer programs to particularly switch between the address spaces of the computer programs by traversing the abstract syntax tree. Edwards in view of Radigan in particular simply teaches, discloses, and suggests that a computer program can be represented by an abstract syntax tree. However, the claimed invention is limited to representing *two* computer programs by the same abstract syntax tree. It is this portion of the invention that is nonobvious over Edwards in view of Radigan. Likewise, the claimed invention so represents two computer programs by one abstract syntax tree specifically to switch between the address spaces of the two computer programs by traversal of the tree. It is also this portion of the invention that is nonobvious over Edwards in view of Radigan.

For this reason alone, too, then, the claimed invention is patentable over the cited prior art either alone or in combination.

Conclusion

Applicants have made a diligent effort to place the pending claims in condition for allowance, and request that they so be allowed. However, should there remain unresolved issues that require adverse action, it is respectfully requested that the Examiner telephone Mike Dryja, Applicants' Attorney, at 425-427-5094, so that such issues may be resolved as expeditiously as possible. For these reasons, this application is now considered to be in condition for allowance and such action is earnestly solicited.

Respectfully Submitted,



April 30, 2008
Date

Michael A. Dryja, Reg. No. 39,662
Attorney/Agent for Applicant(s)

Law Offices of Michael Dryja
1474 N Cooper Rd #105-248
Gilbert, AZ 85233
tel: 425-427-5094
fax: 425-563-2098